Manual de Usuario

GM862 Arduino Shield

Rev. 1.2

MCI-MA-0063

MCI Ltda.

Luis Thayer Ojeda 0115. Oficina 402
Santiago, Chile

www.olimex.cl

Tel: +56 2 3339579
Fax: +56 2 3350589

® MCI Ltda. 2010

Atención: Cambios y modificaciones hechas en el dispositivo, no autorizados expresamente por MCI, anularán su garantía.

Código Manual: MCI – MA - 0063

1 INTRODUCCIÓN 4

2 FUNCIONALIDADES 4

3 PARTES DEL DISPOSITIVO 5

4 GUÍA RÁPIDA DE INSTALACIÓN Y PROGRAMACIÓN DE GM862 ARDUINO SHIELD 6

5 CÓDIGO EJEMPLO GM862 ARDUINO SHIELD 8

6 MAPA DE PUERTOS 12

7 HISTORIA DEL DOCUMENTO 12

1 INTRODUCCIÓN

El objetivo de este documento es la descripción de las funcionalidades del GM862 Arduino Shield.

Este documento abarca todas las funcionalidades que permite el shield, también se sugieren procedimiento de configuraciones básicas del modem GM862.

El GM862 Arduino Shield permite a la tarjeta Arduino hacer llamadas telefónicas, enviar mensajes de texto SMS, enviar datos vía GPRS, accionar pines I/O del módulo y mucho más. Con esto se fusiona la gran cantidad de funcionalidades y flexibilidad de los módems GM862 con la facilidad y sencillez de programación de Arduino. Esto posibilita desarrollar proyectos con Arduino en las siguientes áreas:

- Telemetría y telecontrol
- Sistemas de seguridad inteligentes
- Sistemas de venta automática inteligentes
- Sistemas de gestión de automóviles flota

2 FUNCIONALIDADES

El GM862 Arduino Shield principalmente entrega la alimentación adecuada al módem Telit GM862 y adaptación de pines para éste dispositivo para trabajar con Arduino.

Algunas características específicas para éste dispositivo son:

1) Regulación de voltaje a de 5-9 [VDC] a 3.8 [VDC]
2) Consumo máximo de 3 [A]
3) Led indicador de alimentación
4) Led indicador de Status del modem GM862
5) Led configurable por el usuario
6) Botón de encendido/apagado de modem GM862
7) Botón de reset placa Arduino
8) Conexión a Puerto ADC modem GM862
9) Dimensiones: 76 x 54 [mm]

3 PARTES DEL DISPOSITIVO

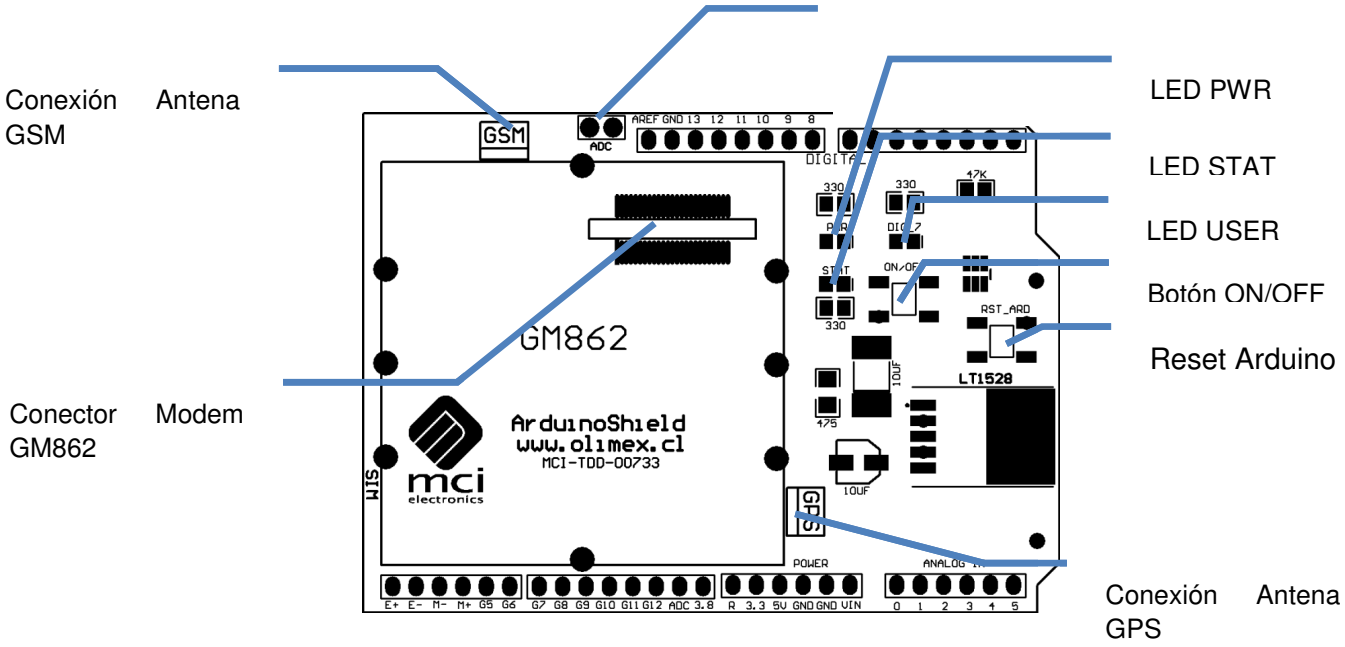

Fig.1 Placa GM862 Arduino Shield

- **Conector Modem GM862**: Conector para modem GM862-GPS
- **LED STAT**: Indicador de funcionamiento de modem GM862
- **LED PWR**: Indicador de alimentación de modem GM862
- **LED USER**: Indicador configurable por el usuario
- **Conector antena GSM**: Conector para antena GSM externa
- **Conector antena GPS:** Conector para antena GPS externa
- **Botón ON/OFF:** Botón de encendido/apagado del modem
- **Botón Reset Arduino**: Resetea placa base Arduino
- **Jumper ADC:** Jumper para setear valor de conversión del ADC del modem GM862 a 0 V.

4 GUÍA RÁPIDA DE INSTALACIÓN Y PROGRAMACIÓN DE GM862 ARDUINO SHIELD

Para la instalación y programación del GM862 Arduino Shield se debe instalar un puerto serial virtual y conectar algunos pines del Shield, de manera de poder tener comunicación con el modem y con el PC a la vez, permitiendo que el procedimiento de depuración del software sea más fluido que utilizando solamente el puerto serial por hardware.

1. Descargar programa de ejemplo aquí.
2. Instalar Shield. Ver fig. 2 para referencia.
    A. Conectar modem GM862
    B. Mover selector a PC-ARD-GM862
    C. Alinear shield de manera de que los pines de conexión de los costados del shield correspondan sean correspondientes con los pines de la placa Arduino
    D. Ensamblar Shield a placa Arduino teniendo cuidado de que TODOS los pines estén conectados
3. Insertar SIM a modem GM862
4. Conectar alimentación 5-9VDC a placa Arduino
5. Conectar puerto USB de Arduino a PC
6. En caso de que no se hayan instalados los drivers FTDI, instalar drivers USB (Descargar aquí)
7. Abrir Arduino Alpha (Descargar aquí)
8. Abrir fuente softSerial_Example.pde (Fuente de ejemplo presentado en capítulo 5). En la carpeta fuente deben estar los archivos NewSoftSerial.cpp y NewSoftSerial.h
9. Editar línea 180 del código de fuente por número de teléfono que desea enviar un SMS
10. Presionar botón Upload en Arduino Alpha
11. Esperar que LED Stat se encienda.
    A. También se puede ver el comportamiento del modem presionando el botón Serial Monitor, configurado a un baudrate de 9600bps.

Al término de ésta secuencia, se envió un mensaje de texto al número indicado en el código de fuente del proyecto.

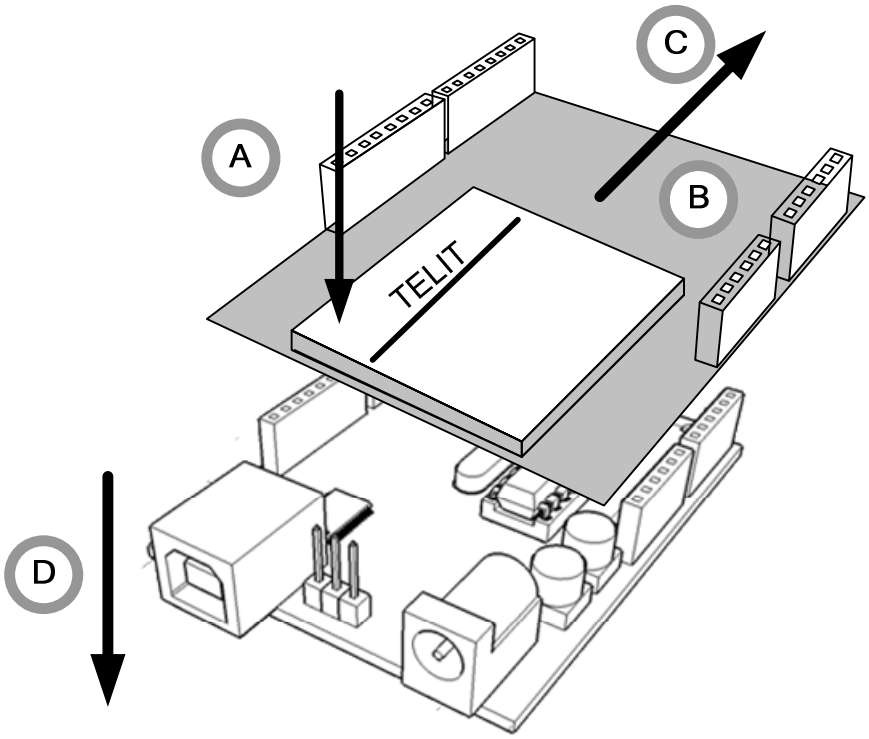

Fig.2 Ensamblaje Placa GM862 Arduino Shield

5 CÓDIGO EJEMPLO GM862 ARDUINO SHIELD

- El siguiente código fue extraído de la biblioteca desarrollada por http://tinkerlog.com y modificado para trabajar con la tarjeta Arduino Duemilanove.
- Éste código envía un mensaje SMS a un número definido por el programador, con un texto pre-definido. Además realiza toda la inicialización y configuración del módem.
- Los archivos fuente pueden se pueden descargar desde la página del autor o desde www.olimex.cl.

```
/*
 * GM862-GPS testing sketch
 * used with Arduino Duemilanove and GM862 Arduino Shield V1.1
 */


#include "NewSoftSerial.h"

#define STATE_REGISTERED 4
#define READ 1
#define txGprs_pin 3
#define rxGprs_pin 2
#define ST_LED_pin 7
#define onOffPin 4
#define BUF_LENGTH 100
NewSoftSerial modem(rxGprs_pin,txGprs_pin);

char buf[BUF_LENGTH];
int state = 0;

byte GM862_requestModem(const char *command, uint16_t timeout, boolean check, char *buf) {

  byte count = 0;
  char *found = 0;
  *buf = 0;
  Serial.println(command);
  modem.flush();
  modem.print(command);
  modem.print('\r');
  count = GM862_getsTimeout(buf, timeout);

  if (count) {
    if (check) {
      found = strstr(buf, "\r\nOK\r\n");
      if (found) {
Serial.println("->ok");
      }
      else {
Serial.print("->not ok: ");
Serial.println(buf);
      }
    }
    else {
      Serial.print("->buf: ");
      Serial.println(buf);
    }
  }
```

```
  else {
    Serial.println("->no respone");
  }
  return count;
}


void GM862_checkNetwork() {
  char buf[BUF_LENGTH];
  char result;
  Serial.println("checking network ...");
  GM862_requestModem("AT+CREG?", 3000, false, buf);
  result = buf[11];

  if (result == '1') {
    state |= STATE_REGISTERED;
  }
  else {
    state &= ~STATE_REGISTERED;
  }
  Serial.println("done");
}


void GM862_init() {
  Serial.println("initializing modem ...");
  char buf[BUF_LENGTH];
  GM862_requestModem("AT", 2000, true, buf);
  GM862_requestModem("AT+IPR=9600", 1000, true, buf);
  GM862_requestModem("AT+CMEE=2", 1000, true, buf);
  GM862_requestModem("AT#BND=1", 2000, true, buf);
  GM862_requestModem("AT+FCLASS=8", 2000, true, buf);
  Serial.println("done");
}


byte GM862_getsTimeout(char *buf, uint16_t timeout) {
  byte count = 0;
  long timeIsOut = 0;
  char c;
  *buf = 0;
  timeIsOut = millis() + timeout;
  while (timeIsOut > millis() && count < (100 - 1)) {

    if (modem.available()) {
      count++;
      c = modem.read();
      *buf++ = c;
      timeIsOut = millis() + timeout;
    }
  }
  if (count != 0) {
    *buf = 0;
    count++;
  }
  return count;
}


void GM862_switchOn() {
  Serial.println("switching on");
  digitalWrite(onOffPin, HIGH);
```

```
  delay(2000);
  digitalWrite(onOffPin, LOW);
  delay(1000);
  Serial.println("done");
}

void GM862_switchOff() {
  Serial.println("switching off");
  digitalWrite(onOffPin, HIGH);
  delay(2000);
  digitalWrite(onOffPin, LOW);
  delay(1000);
  Serial.println("done");
}

void GM862_version() {
  Serial.println("version info ...");
  GM862_requestModem("AT+GMI", 1000, false, buf);
  GM862_requestModem("AT+GMM", 1000, false, buf);
  GM862_requestModem("AT+GMR", 1000, false, buf);
  GM862_requestModem("AT+CSQ", 1000, false, buf);
  Serial.println("done");
}

boolean GM862_isRegistered() {
  return (state & STATE_REGISTERED);
}


void GM862_sendSMS(char *number, char *message) {
  char buf[BUF_LENGTH];
  char cmdbuf[30] = "AT+CMGS=\"";
  Serial.println("sending SMS ...");
  GM862_requestModem("AT+CMGF=1", 1000, true, buf);             // send text sms
  strcat(cmdbuf, number);
  strcat(cmdbuf, "\"");
  GM862_requestModem(cmdbuf, 1000, true, buf);
  modem.print(message);
  modem.print(0x1a, BYTE);
  GM862_getsTimeout(buf, 2000);
  Serial.println(buf);
  Serial.println("done");
}

//-----------------------------------------------
void setup(){
  delay(10000);
   Serial.begin(9600);
   modem.begin(9600);
   //----------------------------------------
   pinMode(txGprs_pin,OUTPUT);
   pinMode(rxGprs_pin,INPUT);
   pinMode(ST_LED_pin, OUTPUT);
   pinMode(onOffPin, OUTPUT);
   digitalWrite(ST_LED_pin, LOW);
   bitSet(MCUCR,PUD);

   // GM862 initialization
   GM862_switchOn();                  // switch the modem on
   delay(4000);                       // wait for the modem to boot
   GM862_init();                      // initialize the GM862
   GM862_version();                   // request modem version info
```

```
        while (!GM862_isRegistered()) {
           delay(1000);
           GM862_checkNetwork();             // check the network availability
        }
     }

     void loop(){
        GM862_sendSMS("6245", "Your Message!");            // Change "6245" by your cellphone
number
        digitalWrite(ST_LED_pin,HIGH);
        delay(10000);
        GM862_switchOff();
        while(1);
     }
```

6 MAPA DE PUERTOS

Mapa de pines de GM862 Arduino Shield

Pin	Nombre/Función
DIGITAL 0	Serial RX
DIGITAL 1	Serial TX
DIGITAL 2	SoftSerial RX
DIGITAL 3	SoftSerial TX
DIGITAL 4	ON/OFF Modem
RESET	RESET Ard
EAR_HF+	Handset ear output, phase +
EAR_HF-	Handset ear output, phase -
MIC_HF-	Handsfree microphone input; phase -, nominal level 3mVrms
MIC_HF+	Handsfree microphone input; phase +, nominal level 3mVrms
GPIO5	Configurable general purpose I/O pin
GPIO6	GP I/O pin or ALARM output pin
GPIO7	GP I/O pin or Buzzer output pin
GPIO8	Configurable general purpose I/O pin
GPIO9	Configurable general purpose I/O pin
GPIO10	Configurable general purpose I/O pin
GPIO11	Configurable general purpose I/O pin
GPIO12	Configurable general purpose I/O pin
ADC	ADC input
3.8V	3.8V regulated voltage

7 HISTORIA DEL DOCUMENTO

Revisión	Fecha	Editado por	Descripción/Cambios
1.0	12 de Agosto 2010	E. Martin	Versión inicial del documento
1.1	22 de Septiembre 2010	E. Martin	Se incluye ejemplo para uso de puerto serial por software
1.2	23 de Noviembre 2010	E. Martin	Actualización de cambios en nueva versión de placa